REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

AD-A228 874

2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
Sept., 1990	First Annual Report

An Evolutionary Approach to Designing Neural Networks

5. FUNDING NUMBERS
(c) AFOSR Contract No.
F 49620-89-K-0005

6. AUTHOR(S)
Stephen Barnard, Senior Computer Scientist
Aviv Bergman, Research Physicist

AFOSR-TR- 90 1069

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
SRI Internatinal
Artificial Intelligence Center
333 Ravenswood Avenue
Menlo Park, CA 94025

8. PERFORMING ORGANIZATION REPORT NUMBER
SRI Project ECU 7929

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
USAF, AFSC
Air Force Office of Scientific Research
Bldg. 410
Bolling AFB, D.C. 20332-6448

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
2305B3

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Distribution unlimited

12b. DISTRIBUTION CODE
Unclassified/unlimited

13. ABSTRACT (*Maximum 200 words*)
One of the most interesting properties of neural networks is their ability to learn appropriate behavior by being trained on examples. Established learning algorithms, which typically work by minimizing error through backpropagation in weight space, tend to get stuck in local optima--a tendency typical of gradient-descent methods applied to nonconvex objective functions. Therefore, for problems of nontrivial complexity these systems must be handcrafted to a significant degree, but the distributed nature of neural network representations make this handcrafting difficult. We are investigating an evolutionary approach to learning that will avoid this problem. This approach simulates a variable population of netowrks which, through processes of mutation, combination, selection, and differential reproduction, converges to a group of netowrks well suited to solving the task at hand. The important components of the approach are a genetic language for coding a large variety of networks, a procedure for constructing networks from these genetic codes, a nondeterministic process for mutating and combining genetic codes, and a function that measures the overall fitness of networks in the context of the task at hand. We use a Connection Machine to exploit the inherent parallelism in these simulations.

14. SUBJECT TERMS
Population Dynamics, Evolution and Coevolution, Unsupervised learning, Adaptation, Neural Networks, Genetic Algorithm.

15. NUMBER OF PAGES
24

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	ul

SRI International

First Annual Report • September 1990

AN EVOLUTIONARY APPROACH TO DESIGNING NEURAL NETWORKS

Stephen T. Barnard, Senior Computer Scientist
Aviv Bergman, Research Physicist
Artificial Intelligence Center

SRI Project 7929

Prepared for:

Air Force Office of Scientific Research
Bolling Air Force Base
Washington, D.C. 20332

Attn: Dr. Alan E. Craig, Building 410

Contract F49620-89-K0005

333 Ravenswood Avenue • Menlo Park, CA 94025-3493 • (415) 326-6200 • FAX (415) 326-5512 • Telex 334486

Contents

1 Introduction and Objectives

Research on an evolutionary approach to designing neural networks that learn was begun at SRI International (SRI) in July 1989 under AFOSR sponsorship (SRI Project 7929, Contract No. F49620-89-K0005). This report describes the research conducted during the first year of the project.

The aim of this program is to design a system that can learn to recognize signals adaptively. That is, the system should learn to respond in a distinctive, repeatable way to those signals to which it has been exposed, should track changes to its signal environment (including possibly the introduction of entirely new classes of signals), and should do these things spontaneously, with no instruction. Adaptative signal recognition should be the result of a self-reorganization of the system in the face of a changing environment. Our hypothesis is that the principles of biological evolution and population genetics provide the basis for such behavior. The processes of variation, selection, and differential reproduction are known to produce in natural populations the kind of emergent behavior we seek to emulate. By simulating these processes on the computer, we hope to observe similar kinds of behavior in artificial systems.

Clearly, this approach requires powerful computing resources. We must simulate statistically significant populations of networks on many inputs over many generations. While such an approach may be impractical on conventional serial computer systems, it has a high degree of implicit parallelism that can be exploited with suitable hardware. Not only can we simulate all members of the population in parallel, but the classes of phenotypes that we study have an internal parallelism that we can exploit. The feedforward perceptrons with hidden units, for example, have a parallel dataflow structure. SRI's Connection Machine provides a nearly ideal computational engine for our work.

Our purpose is not to model biological processes explicitly, but rather to explore a genetic and ecological metaphor of computation. We are interested in investigating this metaphor for two reasons. First of all, adaptive behavior may lead to very general methods of dealing with difficult and ill-defined problems in signal understanding. A system that can learn from experience without explicit training by examples, that can exploit contextual information, and that can modify itself to adapt to possibly radical changes in its input could be useful for difficult problems such as speaker-independent speech recognition. In addition, the inherent parallelism of the evolutionary metaphor, with its emphasis on populations, can lead to effective methods for exploiting the power of parallel computer systems.

2 Status of the Research

2.1 Introduction

The problem we are addressing is a simplified version of the class of problems we will tackle eventually. We describe the general problem, and then describe the simplified version we have investigated this year.

Suppose that we have a system, for the time being regarded as a "black box," that receives as input a *signal vector* of length n, $\mathbf{x} = \langle x_0, \ldots x_{n-1} \rangle$. These signals could be, for example, speech waveforms. The components of $\mathbf{x}$ are real numbers within some limited dynamic range. In practice, since any measurement of a real signal will be uncertain to some degree, we can represent the signal vector with nonnegative integers to some precision b bits. Each possible signal is a point in the n-dimensional metric *signal space*.

Now suppose the system is stimulated only by a much smaller, structured ensemble of signals generated by a few unknown, relatively low-dimensional physical processes, possibly corrupted by noise. They are called *sources*. They could be, for example, a few speakers of

English. There may be considerable variation within a single source, so we should imagine a source to be represented by a subset of the signal space: its *attractor*. The task of the system is to respond distinctively to each source. From looking at a macroscopic feature of the system, we should be able to tell when it has been presented with a source and which source it is.

In the simplified problem we restrict the components of the input vector to binary values ($b = 1$) and restrict the sources to single values (point attractors). Under these assumptions, the system will be learning a subset of the numbers $\{0, \ldots, 2^n - 1\}$. The signal vector can be visualized as the corners of an n-dimensional hypercube, and the response of the system will be to select one of these corners.

2.2 Encoder Populations

Each subsystem is an instantiation of a simple neural network called an *encoder* [1,11] as shown in Figure 1. An n_1-n_2-n_3 encoder has n_1 inputs that feed into n_2 hidden units, which in turn feed into n_3 output units. Each unit computes a weighted sum of the inputs and compares the result with a threshold. If the sum exceeds the threshold, the unit is activated and outputs a one; otherwise, it produces a zero.

Originally, these networks were used to attack the *encoding problem* [11]. Assume that $n_1 = n_3$ and $n_2 = \log_2 n_1$, and that the inputs consist of a single one bit, with all the rest zeros. The position of this bit then represents one of the first n natural numbers. The encoding problem is to learn to encode these numbers into a pattern of $\log n$ bits, and also to learn to decode this $\log n$ bits pattern into an output pattern, usually identical to the input pattern. We, however, are using the population of encoders in quite a different way. Instead of finding a single network that solves the encoding problem for all sources, we want to construct subpopulations of networks that are specialized for encoding different sources.

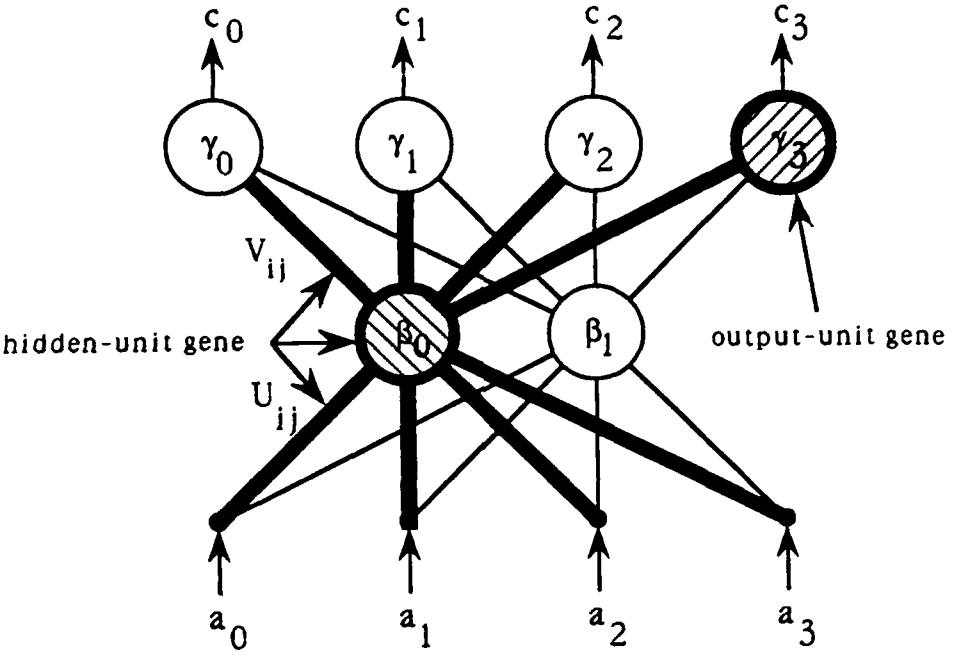

Figure 1: A 4-2-4 encoder.

In general, an encoder is a tuple

$$\xi = [\beta, \gamma, U, V]$$

where $\beta = \langle \beta_0, \ldots, \beta_{n_2-1} \rangle$ and $\gamma = \langle \gamma_0, \ldots, \gamma_{n_3-1} \rangle$ are thresholds for the hidden units and the output units, respectively, and $U = \{u_{ij} | 0 \leq i < n_2, 0 \leq j < n_1\}$ and $V = \{v_{ij} | 0 \leq i < n_2, 0 \leq j < n_3\}$ are weight matrices.

An encoder accepts an n_1-bit input vector **a**, produces an n_2-bit hidden vector **b**, and then produces an n_3-bit output vector **c**. Each unit applies a threshold function

$$\Theta(\phi, s) = \begin{cases} 1 \text{ if } s \geq \phi \\ 0 \text{ otherwise} \end{cases}$$

to the sum of its weighted inputs:

$$b_i = \Theta(\beta_i, \sum_{0 \leq j < n_1} u_{ij} a_j)$$

$$c_j = \Theta(\gamma_j, \sum_{0 \le i < n_3} v_{ij} b_j) \ .$$

It is essential to the genetic algorithm described below that a description of an encoder may be decomposed into parts, called *genes*, in such a way that a new encoder (a child can be constructed with parts from two others (the parents) [7,5]. In part, we have chosen the encoder network for this work because it can be decomposed in a fairly natural way. The genetic structure of an encoder is illustrated in Figure 1. Each encoder has n_2 hidden-unit genes and n_3 output-unit genes. The hidden-unit genes are the more complex of the two types. The ith hidden-unit gene of an encoder ξ consists of the hidden-unit threshold β_i, a vector of input weights $\langle u_{ij} | 0 \le j < n_1 \rangle$, and a vector of hidden-unit weights $\langle v_{ij} | 0 \le j < n_3 \rangle$. The jth output-unit gene consists simply of the output-unit threshold γ_j.

The system consists of a population of N encoders

$$\Xi = \{\xi_k, 0 \le k < N\}$$

with, in general, different thresholds and weights. We always have $n_1 = n_3$ and typically, but not necessarily, $n_2 = \log_2 n_1$. Every encoder in the population is presented simultaneously with the same input vector, and tries to reconstruct the input. Success is measured by a *fitness function* [3,8]

$$f_k(\mathbf{a}) = - \sum_{0 \le j < n} |a_i - c_i| \ .$$

Note that fitness is simply the negative of the Hamming distance between the input and the output vectors. The idea behind the genetic algorithm described below is to increase the frequency of genes and combinations of genes in Ξ by selection, thereby causing the population to learn to encode the inputs it sees most frequently.

2.3 A Genetic Algorithm

Genetic algorithms can be effective for exploring large design spaces [5,7]. The essential idea is to simulate many generations of populations of individual subsystems, with each generation produced from previous generations by selection and differential reproduction [3,4,6,10]. Each individual is graded by a fitness function that is intended to measure its performance on one or more instances of a problem. Those individuals that are most fit are selected and then a set of new subsystems is created by applying genetic operators to the descriptions of the selected individuals. Commonly used genetic operators are called *crossover* and *mutation*, modeled after similar processes that drive biological evolution [2,5,7]. Although the concepts behind genetic algorithms are very general, there are inevitably a wide variety of parameters, reproduction schemes, representations, and so on that could be used. Part of the aim of this preliminary work is to understand the consequences of and interactions among these choices.

Our genetic algorithm consists of an initialization,

$$\Xi \leftarrow \Xi^0$$

followed by an iteration of the generation operator, $\mathcal{G}$:

$$\Xi \leftarrow \mathcal{G}(\Xi, \mathbf{a}^t),\ t = 0, 1, \ldots$$

In the initialization step, a population of at least $N = 4096$[1] encoders with n inputs and m hidden units is created. All thresholds and weights are chosen from a uniform random distribution over the interval $[-1, 1]$. Initially, all of the members of Ξ are marked as alive and are assigned an age chosen from a random distribution of integers in the range $[0, \ldots, age_{max} - 1]$. Only those encoders marked as alive, denoted by Ξ_a, are active and available for input,

[1] We use a Connection Machine with 4096 processors for our simulations. N can be larger than 4096, but must be a power of 2.

selection, and reproduction. All encoders that are not alive are treated as available space for the next generation. The age of ξ is an integer indicating the number of generations for which ξ has been continuously alive.

The generation function $\mathcal{G}$ is defined as the following sequence of steps:

$$\begin{aligned}
\Omega &\leftarrow \mathit{select}(f, \Xi, \mathbf{a}) \\
\Omega^* &\leftarrow \mathit{reproduce}(\Omega) \\
\Xi &\leftarrow \mathit{insert}(\Omega^*, \Xi) \\
\Xi &\leftarrow \mathit{age}(\Xi) \\
\Xi &\leftarrow \mathit{kill}(\Xi)
\end{aligned}$$

These steps can be performed in several ways, but each step has the basic characteristics outlined below, in Section 2.4.

Selection: $\Omega \leftarrow \mathit{select}(f, \Xi, \mathbf{a})$

An input bit vector, $\mathbf{a}$, is chosen and presented to the system. The input can be selected in a variety of ways. The simplest is to select the vector from a set of sources according to some prior probability distribution. Input vectors can be degraded with noise by inverting bits with some probability. Inputs can also be chosen randomly from the set of 2^n possible inputs with some specified frequency. All living encoders are ranked by fitness and a subset Ω of the most fit is selected. The size of Ω could be determined dynamically by a threshold on fitness. Instead, in this preliminary investigation, we set the size of Ω as a fixed proportion of the size of Ξ (usually 1/16).

Reproduction: $\Omega^* \leftarrow \mathit{reproduce}(\Omega)$

Every member of Ω is paired at random with another member of Ω (possibly itself), which is called its mate. The pairs are combined to produce a fixed number of children. The combination is performed by applying two genetic operators, crossover and mutation. In the crossover operation, every child's gene is selected from one or the other parent with

probability 1/2, a process called *free recombination* [6,9]. In the mutation operation, every gene constituent, whether a weight or a threshold, is replaced by a random value with some probability of mutation μ, which is usually quite low.

Insertion: $\Xi \leftarrow insert(\Omega^*, \Xi)$

A random number $k \in \{0, \ldots, N-1\}$ is generated for every child in Ω^*. If ξ_k is not alive, the child is inserted into Ξ at that location, is marked as alive, and is assigned an age of zero. If more than one child tries to occupy the same location, one child is chosen at random.

Aging: $\Xi \leftarrow age(\Xi)$

The ages of all living encoders are increased by 1.

Death: $\Xi \leftarrow kill(\Xi)$

Every encoder whose age is greater than age_{max} is marked as not alive. Its space in Ξ then becomes available for the children in the next generation.

2.4 Results

When interpreting the performance of the system, we consider only those encoders that can reconstruct their outputs perfectly. These are said to respond to the input; that is, $r_k(\mathbf{a}) = 1$, where

$$r_k(\mathbf{a}) = \max(0, 1 + f_k(\mathbf{a})) \, .$$

We want many networks to respond to the sources, few or none to respond to nonsource signals, and different subpopulations to respond to each different source.

Two measures of the effectiveness of the system depend on computing the probability distribution $P(\mathbf{a}|r)$, which is the probability that the signal is $\mathbf{a}$ given that a randomly chosen encoder is responding. This distribution is computed assuming no prior knowledge of the frequency of occurrence of the source. Therefore, using a uniform (maximum entropy)

distribution of priors

$$P(\mathbf{a}) = \frac{1}{2^n}$$

and writing the probability of an encoder responding to $\mathbf{a}$ as

$$P(r|\mathbf{a}) = \frac{\sum_k r_k(\mathbf{a})}{N} ,$$

and the probability of an encoder responding to any signal as

$$P(r) = \frac{\sum_{\mathbf{x}} \sum_k r_k(\mathbf{x})}{N 2^n} ,$$

we use Bayes's Rule to determine the desired distribution:

$$P(\mathbf{a}|r) = \frac{P(r|\mathbf{a})P(\mathbf{a})}{P(r)} ,$$

or

$$P(\mathbf{a}|r) = \frac{N \sum_k r_k(\mathbf{a})}{\sum_{\mathbf{x}} \sum_k r_k(\mathbf{x})} .$$

Ideally, this distribution should be identical to the prior probability $P(\mathbf{a})$ after many generations.

We can compute the entropy of $P(\mathbf{a}|r)$

$$S = -\sum_{\mathbf{x}} P(\mathbf{x}|r) \log_2 P(\mathbf{x}|r)$$

to summarize the degree of organization of the system in terms of the uncertainty associated with its response. We can also compute the correlation between $P(\mathbf{a}|r)$ and some prior model distribution $P_M(\mathbf{a})$ from which the sources were chosen:

$$C = \frac{\sum_{\mathbf{x}} (P(\mathbf{x}|r) - \overline{P(\mathbf{x}|r)})(P_M(\mathbf{x}) - \overline{P_M(\mathbf{x})})}{\sqrt{\sum_{\mathbf{x}} (P(\mathbf{x}|r) - \overline{P(\mathbf{x}|r)})^2} \sqrt{\sum_{\mathbf{x}} (P_M(\mathbf{x}) - \overline{P_M(\mathbf{x})})^2}} .$$

The first three experiments described below use entropy and correlation to examine the evolution of the system under different conditions. Because the time required to compute

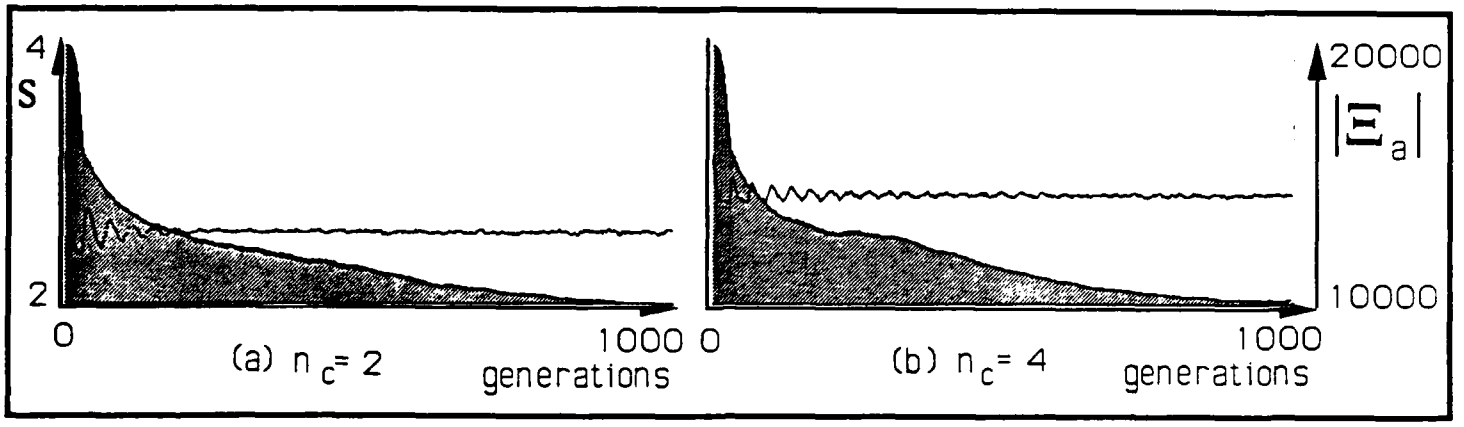

Figure 2: Typical behavior (no mutation)

$P(\mathbf{a}|r)$ grows exponentially with the length of the input vector, n, these experiments were done only on small 4-2-4 encoders. The fourth experiment examines the behavior of the system when n is larger and, in particular, when the number of possible inputs greatly exceed the size of the population. Finally, the fifth experiment examines whether the population becomes specialized to the sources.

2.4.1 Experiment 1: Typical Behavior (no mutation)

The first experiment examines the typical behavior of a population of 16K 4-2-4 encoders with no mutation ($\mu = 0$). The inputs were chosen at random with equal frequency from a set of four sources. Figure 2 shows the entropy of $P(\mathbf{a}|r)$ over 1000 generations when the maximum number of children n_c is 2 and 4 ((a) and (b), respectively). Also shown is the size of the population that is living.

In both cases the entropy eventually drops to the ideal value of $\log_2 4 = 2$, which is the entropy of the model distribution. The correlation with the model distribution (not shown) is very nearly 1 after only about 20 generation. The fraction of the population that is living fluctuates at first, but eventually approaches some limit, which is greater for the $n_c = 4$ case.

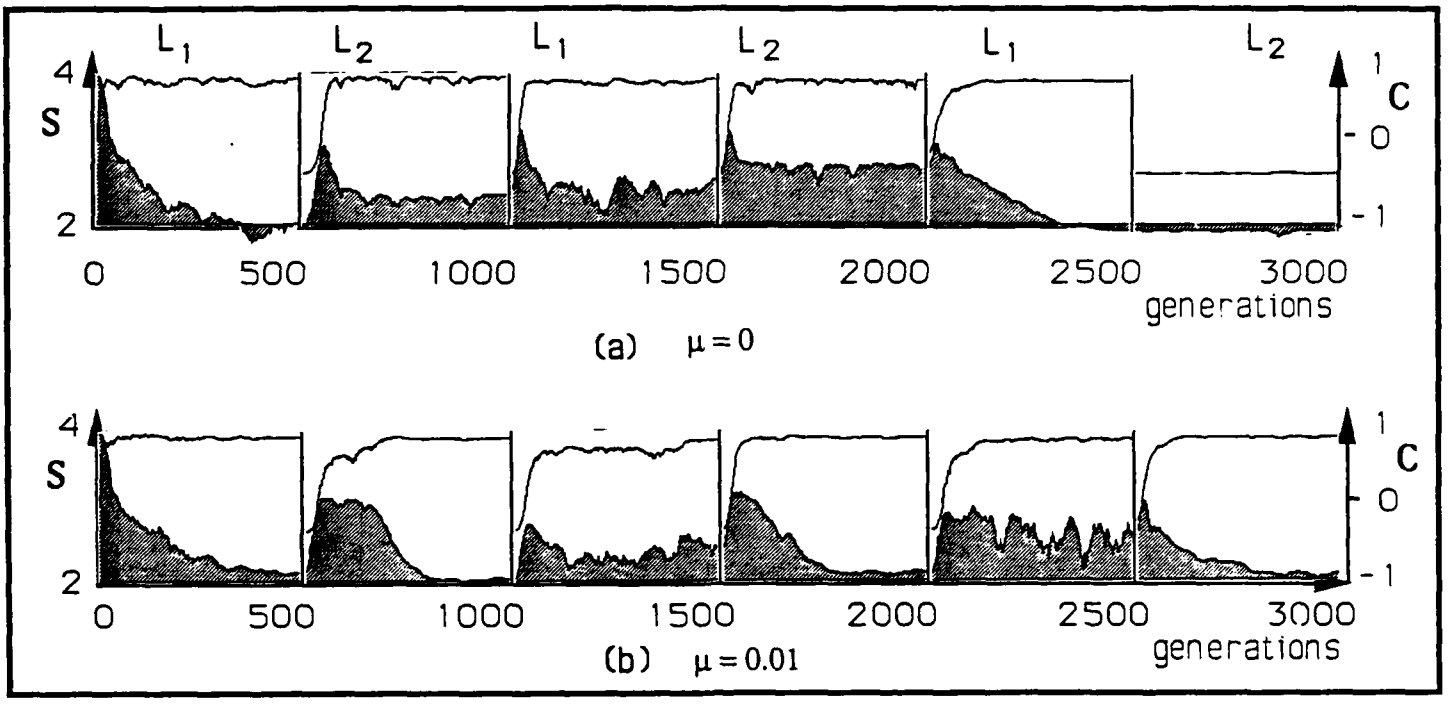

Figure 3: Changing environment

2.4.2 Experiment 2: Changing Environment

The previous simple experiment illustrated that adaptation can occur without mutation, relying only on the crossover operation. This experiment shows that mutation is essential in a more challenging problem. Figure 3 shows the entropy and the correlation measures when the system is successively stimulated with two different sets of four signals, L_1 and L_2. Two cases are shown: $\mu = 0$ and $\mu = 0.01$. The interesting feature of this experiment is that in the first case, $\mu = 0$, the system "collapses" into an irreversible condition of total insensitivity on the third presentation of the set L_1. The entropy drops to zero, indicating that the system can respond to no signals (or possibly to only one), and the correlation with the model distribution drops effectively to zero. Apparently, the successive presentations and epochs of selection have eliminated variation in Ξ. Selection for L_1 eliminates genes effective for L_2, selection for L_2 eliminates genes effective for L_1, and so on, until by the third presentation of L_1, Ξ has been so depleted that it cannot adapt.

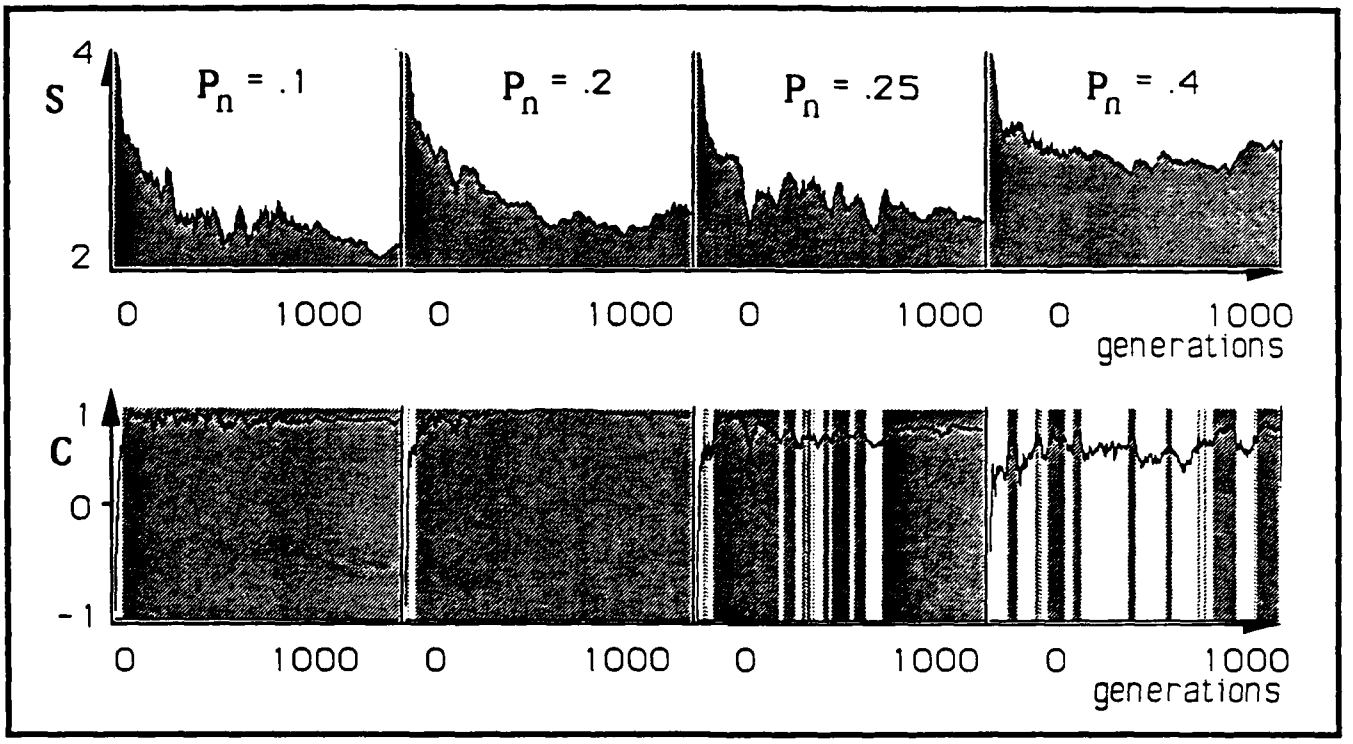

Figure 4: Effects of noise

In the case of $\mu = 0.01$ this does not happen. Even this low rate of mutation is sufficient to maintain adequate variation in Ξ. The crossover operation is effective for making large jumps though the space of genotypes, while mutation is effective as a continual source of variation.

2.4.3 Experiment 3: Effects of Noise

Experiment 3 examines the effects of noise in the input. The population size is 4K, the encoders are 4-2-4, four different sources are used with equal probability, $\mu = 0.01$, $n_c = 4$, and $age_{max} = 30$. Each encoder is presented with an input vector, selected from the four sources, but each vector has a probability P_n of having (at least) one bit changed at random. All encoders receive input from the same source, but the inputs are corrupted by noise independently, so that any two encoders may see different signals. Figure 4 shows four cases:

$P_n = 0.1, 0.2, 0.25, 0.4$. Entropy is shown above and correlation below. The shaded portions of the correlation graphs indicate when the system is working, in the sense that the four signals of highest probability are identical to the sources. The system performs well up to $P_n = 0.2$ but degrades quickly for higher noise levels.

2.4.4 Experiment 4: Large n

To test the system on a larger problem, and in particular on a problem in which the number of possible signals greatly exceeds the size of Ξ, we performed a simulation with 16-4-16 encoders and eight sources. As in the previous simulation the population size is 4K, $\mu = 0.01$, $n_c = 4$, and $age_{max} = 30$. Because the number of possible inputs is $2^{16} = 64K$ it is not practical to compute the complete distribution $P(\mathbf{a}|r)$, especially not for every generation. Instead, we let the system run for 4,000 generations and then counted the number of encoders that responded averaged over all eight sources, which was 488.5, and the average number of encoders that responded averaged over 1000 randomly chosen signals, which was 0.13.

2.4.5 Experiment 5: Specialization

The last experiment examines whether the population divides into disjoint subpopulations specialized for the sources. Suppose we have s sources with R_i being the subpopulation of encoders that respond to source i. The following equation gives a normalized measure of the overlap between two subpopulations:

$$O_{ij} = \frac{|R_i \cap R_j|}{|R_i \cup R_j|} \quad 0 \leq i, j < s \; .$$

Ideally, O_{ii} should be one if $i = j$ and zero otherwize for complete specialization. Figure 5 shows matrices of overlap measures for four cases. When we adapt 4-2-4 encoders to only two sources, shown in Figure 5 (a), no specialization occurs at all: nearly every encoder

2 sources overlap matrix (for 4-2-4 encoders)

	S1	S2
S1	1.0	0.99
S2	0.99	1.0

(a)

4 sources overlap matrix (for 4-2-4 encoders)

	S1	S2	S3	S4
S1	1.0	0.99	0.99	0.0
S2	0.99	1.0	0.99	0.0
S3	0.99	0.99	1.0	0.0
S4	0.0	0.0	0.0	1.0

(b)

7 sources overlap matrix (for 4-2-4 encoders)

	S1	S2	S3	S4	S5	S6	S7
S1	1.0	0.22	0.62	0.53	0.0	0.0	0.0
S2	0.22	1.0	0.43	0.33	0.0	0.0	0.0
S3	0.62	0.43	1.0	0.69	0.0	0.0	0.0
S4	0.53	0.33	0.69	1.0	0.0	0.0	0.0
S5	0.0	0.0	0.0	0.0	1.0	0.99	0.99
S6	0.0	0.0	0.0	0.0	0.99	1.0	0.99
S7	0.0	0.0	0.0	0.0	0.99	0.9	1.0

(c)

10 sources overlap matrix (for 16-2-16 encoders)

	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10
S1	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
S2	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
S3	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
S4	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.66
S5	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0
S6	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0
S7	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0
S8	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0
S9	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0
S10	0.0	0.0	0.0	0.66	0.0	0.0	0.0	0.0	0.0	1.0

(d)

Figure 5: Specialization

that responds to one source also responds to the other. When we adapt the same system to four sources (b) or seven sources (c), there is some specialization, with relatively more specialization occurring when there are more sources. Finally, when we adapt a system of 16-2-16 encoders to ten sources, Figure 5 (d), the specialization is nearly perfect, with only two subpopulations having a significant degree of overlap.

2.5 Summary of Accomplishments

For the encoding problem, the evolutionary algorithm exhibits effective adaptation. Differential reproduction amplifies the frequency of selected genes and leads to the emergence of a population that is progressively more fit. In our model, free recombination (crossover) seems to be the primary means of adaptation. Two relatively fit parents clearly have a better-than-average chance of producing more fit offspring. Mutation, on the other hand, has only an average chance of producing an offspring that is more fit, regardless of the parents' fitness. However, by itself free recombination causes a progressive loss of information: those genes that are amplified replace others that are lost forever. This loss of diversity in the gene pool is disastrous if the ensemble of sources changes, as demonstrated in Experiment 2. The mutation operator continuously injects diversity into the gene pool, thereby preventing the system from becoming trapped in a low-diversity dead end.

Our approach differs from some genetic-algorithm and neural-network approaches in a fundamental way. We do not seek an individual encoder that is "most fit" overall; instead, we seek subpopulations of networks that have specialized their responses to particular sources. The response of the system is an aggregate, macroscopic feature of the individual responses of a large population of individual, interacting subsystems. We view fitness as a very general concept: simply a measure of the similarity between the input and the output. Rather than being built in to the fitness function, the evolutionary trend toward specialization is instead

an emergent property of the population as a whole, and a consequence to the informational bottleneck in the encoders. Unlike the more standard optimization methods for designing systems, this method results in subpopulations that resemble species adapted to different ecological niches that are determined by the sources.

We would like to simulate populations with more diverse features, such as variable sizes, reproduction rates, age limits, and mutation rates. Currently, these properties are global to all encoders, but they could be variable, inherited properties, represented as "modifier genes" attached to the basic encoder genotype. We speculate that this process will lead to more interesting adaptation because it will create more niches for adaptation to fill. For example, one can imagine relatively large, scarce, long-lived encoders specializing on complex sources that appear infrequently or change slowly or relatively small, numerous, short-lived, and perhaps highly mutable encoders specializing on common, simple sources. A further interesting possibility is the coevolution of interacting populations in symbiotic or parasitic relationships [10].

We are changing the input representation of the more general case of b-bit samples so that we can investigate applications to real, physical sources. Whether the approach can be extended to more complex sources than point attractors is an open question. To do so, the basic encoder representation may have to be extended to a more elaborate, dynamic network. Instead of an encoder, we may need a generator whose internal state allows it to recognize and mimic (i.e., predict) a sours with a low number of dimensions.

3 Publications and Presentations

A paper by Stephen T. Barnard and Aviv Bergman will be published in the *Proceedings of Parallel Problem Solving from Nature*, a workshop held in Germany, on October 1990. Aviv

Bergman also participated in the international workshop on *Evolution and Complex System*, in Torino, Italy, on July 1990. This workshop included fruitful discussion among several of the world's top researchers into complex systems and evolution.

4 Personnel

AVIV BERGMAN

Research Physicist
Artificial Intelligence Center
Computing and Engineering Sciences Division

SPECIALIZED PROFESSIONAL COMPETENCE

Address block location using high resolution multispectral images, including development of image processing, signature detection, classification and ranking algorithms

Statistical image and range data analysis, image-processing and pattern recognition using classical techniques, simulated annealing and neural-networks methods

Bar code detection and decoding algorithms

2-D and 3-D image processing and understanding for arc-welding robots. Digital filters, heuristics and geometrical analysis of stereo and laser scanner data and development of the corresponding range sensors

The application of massively parallel machines (Connection Machine) and algorithms (neural-networks like) for optimization problems, namely, stereo vision (including occlusions boundaries), and image segmentation

Simulated Evolution, and the acquisition of learning and computation capabilities in massively parallel architecture using population genetics theory

REPRESENTATIVE RESEARCH AT SRI (since 1985)

Analysis of images of mail pieces

Supervised and unsupervised classification technique

Classification using neural networks

Bar code detection and decoding algorithms

Noise-tolerant range image analysis for autonomous navigation

Robot manipulation grasping estimation

OTHER PROFESSIONAL EXPERIENCE

Spline functions and predictions of 1-D and 2-D signals

Medical applications of signal processing

Image Processing, Department Manager, Elco Robotics

Population genetics and evolutionary theory

ACADEMIC BACKGROUND

PhD. studies at Stanford University, Biological Science Department. 1988 -

Department of Physics, Weizman Institute, 1975-1976

Department of Physics and Electrical Engineering. the Technion, Haifa, Israel, 1972-1975

PUBLICATIONS

"An Optical Height Measuring System for Operation in a Noisy Environment," Israeli Patent No. 71948

Authored or coauthored numerus papers and reports on evolution theory, neural-networks, image analysis. and robotics

PROFESSIONAL ASSOCIATIONS AND HONORS

IEEE., AAAI., AAAS., INNS., BBS.

STEPHEN T. BARNARD

Senior Computer Scientist
Artificial Intelligence Center
Computing and Engineering Sciences Division

SPECIALIZED PROFESSIONAL COMPETENCE

Artificial intelligence, machine vision, parallel computing

REPRESENTATIVE RESEARCH AT SRI (since 1979)

Interpretation of perspective images of natural scenes, shape from contour under perspective, application of machine vision techniques to industrial problems, perception for mobile robots, theory and modeling of stereo vision, neural-network models for vision, implementations on the Connection Machine superconputer

OTHER PROFESSIONAL EXPERIENCE

Industrial lectureship, Computer Science Dept., Stanford U., CA

ACADEMIC BACKGROUND

B.S., Mechanical Engineering, Case Western Reserve U. (1969); M.S., Computer Science, U. of Minnesota (1976); Ph.D., Computer Science, U. of Minnesota (1979)

PARTIAL LIST OF PUBLICATIONS

"Stochastic Stereo Matching on the Connection Machine," *Proceeding of the Fourth International Conference on Supercomputing*, Santa Clara, CA (May 1989)

"Stochastic Stereo Matching over Scale," *International Journal of Computer Vision*, vol. 3(1), 1989

"Stereo Vision," *Encyclopedia of Artificial Intelligence*, John Wiley and Sons (1987) (with M. Fischler)

"A Stochastic Approach to Stereo Vision," *Proceedings of the AAAI-86*, Philadelphia (August 1986)

"Choosing a Basis for Perceptual Space," *Computer Vision, Graphics, and Image Processing*, vol. 29, (1983)

"Interpreting Perspective Images," *Artificial Intelligence*, vol. 21, (1983)

"Computational Stereo," *ACM Computing Surveys*, (December 1982) (with M. Fischler)

"Modeling and Using Physical Constraints in Scene Analysis," *Proceedings of the AAAI-82*, Pittsburgh (August 1982) (with A. Pentland)

"Lower-Level Estimation and Interpretation of Visual Motion," *Computer*, (August 1981) (with W. Thompson)

"Disparity Analysis of Images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. PAMI-2 (July 1980) (with W. Thompson)

Author, "The Image Correspondence Problem," U. of Minnesota Ph.D. thesis (November 1979)

PROFESSIONAL ASSOCIATIONS

Member, American Association for the Advancement of Science
Member, American Association for Artificial Intelligence

References

[1] Ackley, D. H., G. E. Hinton, and T. J. Sejnowski, "A learning machine for Boltzman Machines," *Cognitive Science, 9*, 147-169, 1985.

[2] Bergman, A., and M.W. Feldman, "More on selection for and against recombination," *Journal for Theoretical Population Biology*, 1990.

[3] Bergman, A., and M. Kerszberg, "Breeding intelligent automata," *IEEE First Annual Conference on Neural Networks*, San Diego, June 1987.

[4] Ewens, W.J., *Mathematical Population Genetics*, Springer-Verlag, 1979.

[5] Goldberg, David E., *Genetic Algorithms*, Addison-Wesley, 1989.

[6] Hartl, D.L., and A.G. Clark, *Principles of Population Genetics*, (2nd edition), Sinauer Associates, Inc. Publishers, 1989.

[7] Holland, J.H., *Adaptation in Natural and Artificial Systems*, Ann Arbor : University of Michigan Press, 1975.

[8] Kerszberg, M., and A. Bergman, "The evolution of data processing abilities in competing automata," in *Computer Simulation in Brain Science*, ed. Rodney M.J.Cotterill, pp. 249-259, Cambridge, U.K.: Cambridge University Press, 1988.

[9] Packard, N., "Evolving bugs in a simulated ecosystem," in *Artificial Life*, ed. Christopher G. Langton, Addison Wesley Publishing Company, 1989.

[10] Roughgarden, J., *Theory of Population Genetics and Evolutionary Ecology: An Introduction*, Macmillan Publication Co., Inc., 1979.

[11] Rumelhart, D. E., G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," in *Parallel Distributed Processing*, vol. 1, ed. D. E. Rumelhart and J. L. McClelland, Cambridge, Massachusetts: The MIT Press 1986.

[12] Bateson, P.P.G. "Gene, evolution, and learning," in *The Biology of Learning*, eds. P. Marler and H.S. Terrace, pp. 75-88: Dahlem Konferenzen 1984, Springer-Verlag, 1984.

[13] Changeux, J.P., T. Heidmann, and P. Patte "Learning by selection," in *The Biology of Learning*, eds. P. Marler and H.S. Terrace, pp. 115-133: Dahlem Konferenzen 1984, Springer-Verlag 1984.

[14] Gold, J.L. and P. Marler"Ethology and the natural history of learning," in *The Biology of Learning*, eds. P. Marler and H.S. Terrace, pp. 47-74: Dahlem Konferenzen 1984, Springer-Verlag 1984.

[15] Maxwell, T., C. Lee Giles, and Y. C. Lee, "Generalization in neural networks: The contiguity problem," *Proceedings of the IEEE International Conference on Neural Networks*, San Diego, CA, June 1987.

[16] Edelman, G.M. "Group selection and phasic reentrant signaling: A theory of higher brain function," in *The Mindful Brain* (with V. B. Mountcastle), Cambridge, Massachusetts: M.I.T. Press, 1987.

[17] Reeke, G.N, and G.M. Edelman, "Selective Networks and Recognition Automata," *Annals of the New York Academy of Science,* Vol. 429, 181-201, 1984.